Errors + Graceful Failure

When AI makes an error or fails, things can get complicated. This chapter covers:

- When do users consider low-confidence predictions to be an “error”?
- How will we reliably identify sources of error in a complex AI?
- Does our AI allow users to move forward after an AI failure?

Want to drive discussions, speed iteration, and avoid pitfalls? Use the worksheet.

What’s new when working with AI

As users interact with your product, they’ll test it in ways you can’t foresee during the development process. Misunderstandings, false starts, and impropriety will happen, so designing for these cases is a core component of any user-centered product. Errors are also opportunities. They can support faster learning by experimentation, help establish correct mental models, and encourage users to provide feedback.

There are some great guides for communicating errors that still apply when creating AI-driven features and products, including these Error Messaging Guidelines from the Nielsen Norman Group.

Key considerations for dealing with errors in AI-driven systems:

① **Define “errors” & “failure”**. What the user considers an error is deeply connected to their expectations of the AI system. For example, a recommendations system that’s useful 60% of the time could be seen as a failure or a success, depending on the user and the purpose of the system. How these interactions are handled establishes or corrects mental models and calibrates user trust.

② **Identify error sources**. With AI systems, errors can come from many places, be harder to identify, and appear to the user and to system creators in non-intuitive ways.

③ **Provide paths forward from failure**. AI capabilities can change over time. Creating paths for users to take action in response to the errors they encounter encourages patience with the system, keeps the user-AI relationship going, and supports a better overall experience.

① Define “errors” & “failure”

Integrating AI into your product means establishing a relationship that changes as the user interacts with the AI system. Users will likely come to expect that interactions are curated to their tastes and behavior, and that experiences will differ from user to user.

What defines an error, then, can also differ from user to user based on their expertise, goals, mental models , and past experience using the product.

For example, if someone assumes that an AI-driven music recommendation system is only using songs indicated as “favorites” to provide recommendations, then they might consider it an error if the system recommends a genre outside of what’s in their favorites list. Someone else who assumes recommendations are based on a wide range of factors might not consider this an error. There’s more information on fostering a good user understanding of the system’s capabilities, limitations, and interaction model in the chapter on Mental Models.

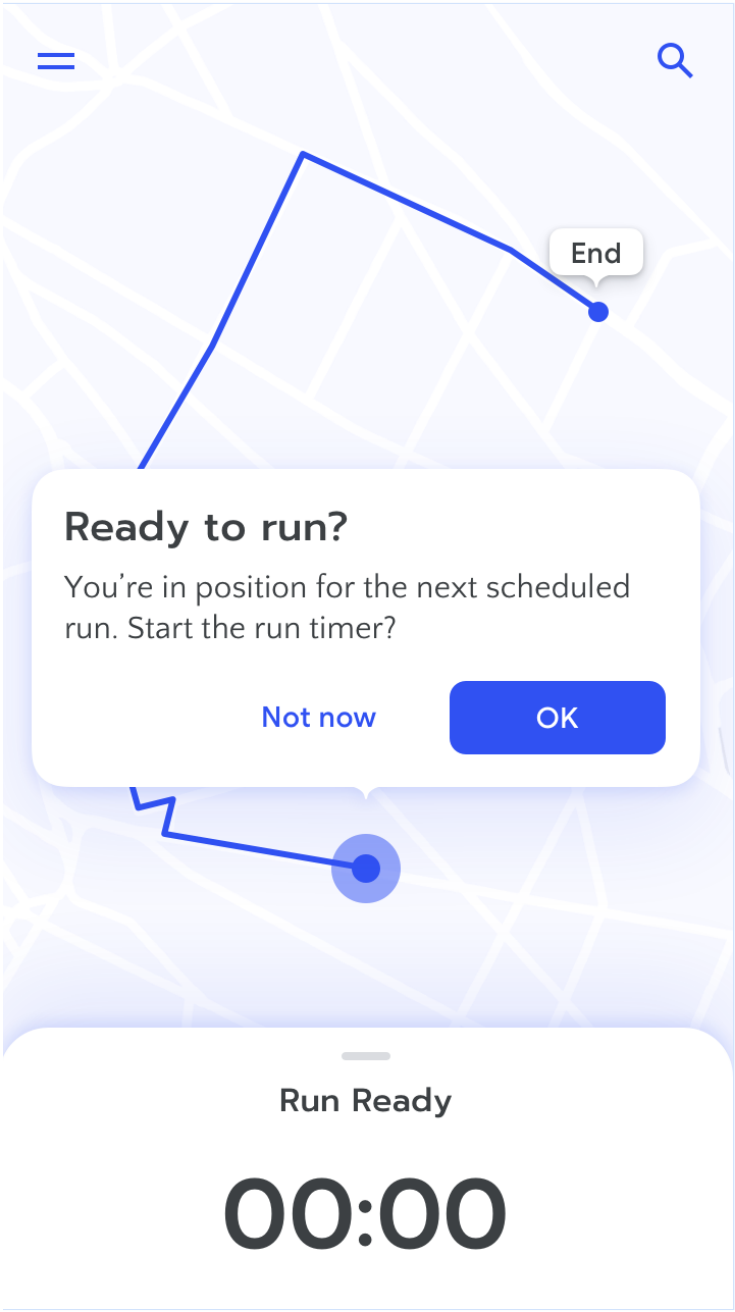
Identify user, system, and context errors

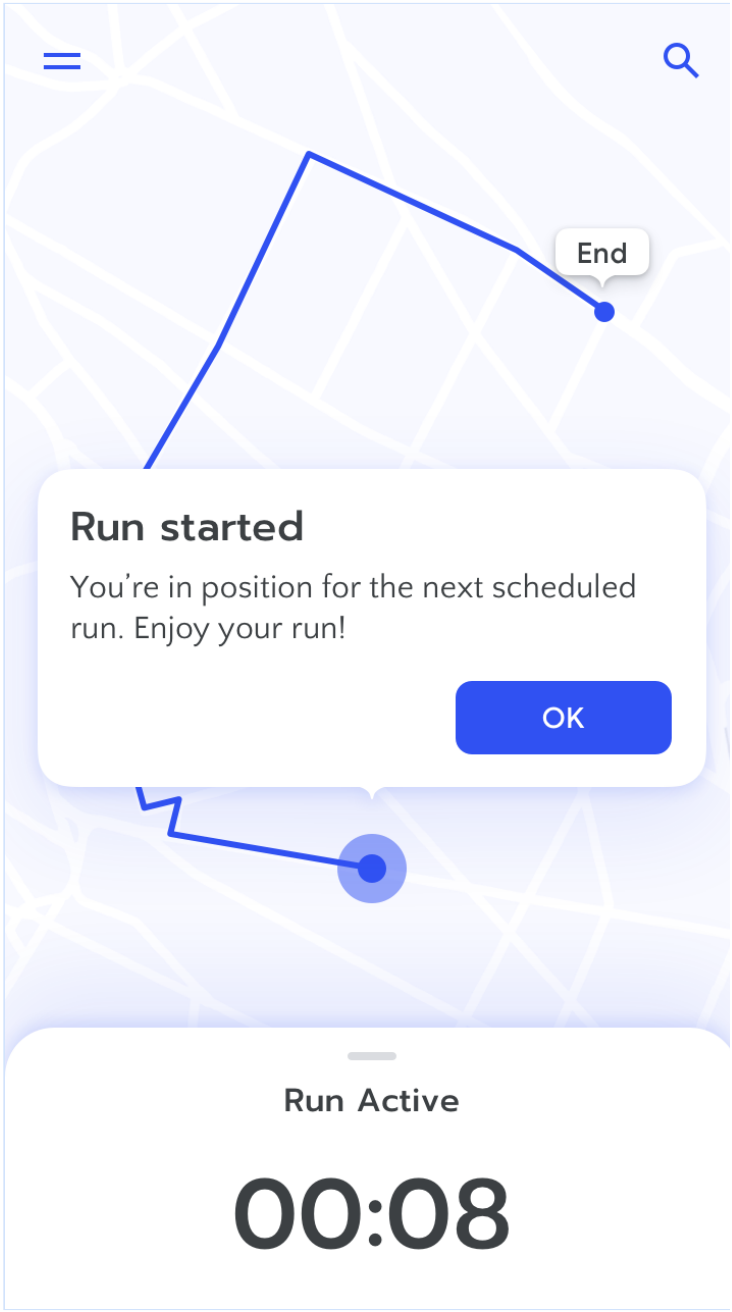
In non-AI systems, “user errors” are typically defined from the point of view of the system designers, and the user is blamed for “misuse” resulting in errors. “System errors” on the other hand, are typically defined from a user’s point of view: users blame the system designer for a product that’s not flexible enough to meet their needs. In AI systems, users can encounter a third type of error, based on the system’s assumptions about the user. These are context errors .

Context errors are instances in which the AI system becomes less useful through making incorrect assumptions about what the user wants to do at a certain time or place. As a result, users might be confused, fail at their task, or abandon the product altogether. Context can be related to individual lifestyle or preferences, or patterns can be linked to wider cultural values.

For example, if someone using a recipe suggestion app frequently rejects recommendations in the evening, this pattern should signal to the product team that there may be a persistent reason why. If the user works a night shift, they might simply prefer breakfast recommendations before they head to work. Whereas if all of the users in a certain group consistently rejected meat-based suggestions during a certain time of year, that might be linked to a cultural value or preference that your team hasn’t accounted for.

To prevent or correct for context errors, you'll need to look at the signals that the AI uses to make assumptions about the user's context and evaluate if they are incorrect, overlooked, undervalued, or overvalued.





Aim for

Provide proactive recommendations when AI has high confidence about the user's context. Learn more

Avoid

Don't act on assumptions. Doing so can lead to context errors.

Context errors and other errors can be thought of as an interaction between user expectations and system assumptions. Here's a breakdown of how user expectations and system expectations interact and cause errors.

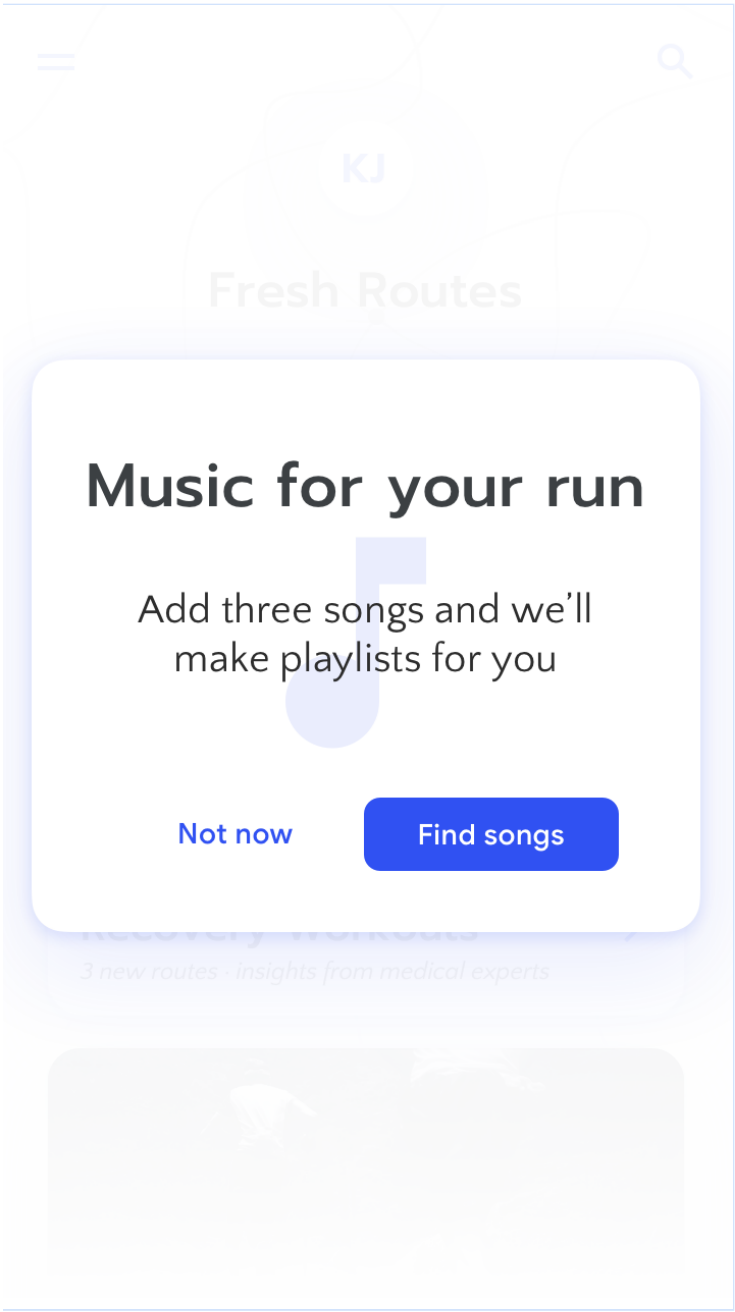
Categorize user-perceived errors

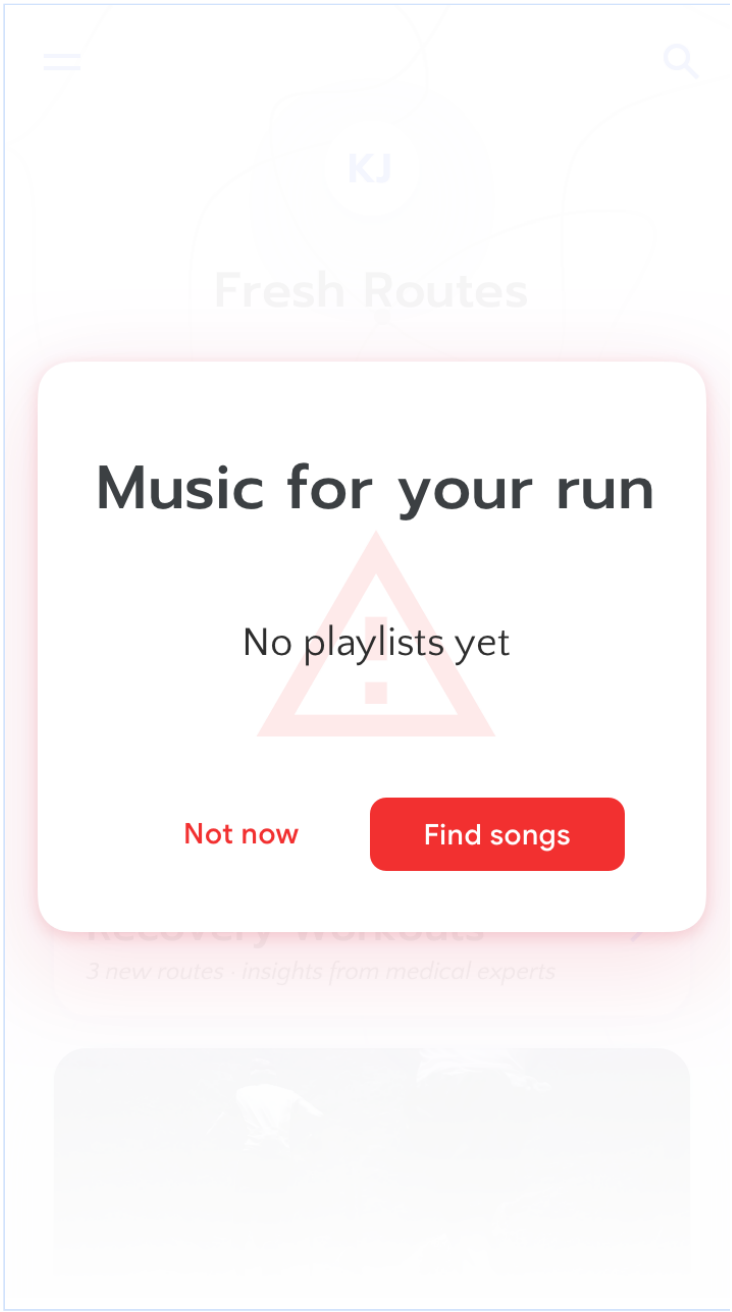
Context errors. These are often true positives : the system is "working as intended," but the user perceives an error because the actions of the system aren't well-explained, break the user's mental model, or were based on poor assumptions. For example, if a friend's flight confirmation email creates an event on your calendar, but you don't want or need it.

Addressing these errors depends on their frequency and severity. You could change the way the system functions to better align with user needs and expectations. Or, adjust your product's onboarding to establish better mental models and shift the user's perception of these situations from errors to expected behaviors.

Failstates. Your system can't provide the right answer, or any answer at all due to inherent limitations to the system. These could be true negatives : the ML is correct in discerning that there's no available output for a given input, but according to users, there should be. For example, if an image recognition app can identify many different kinds of animals, but a user shows the app a photo of an animal that wasn't in the training dataset and therefore is not recognized, that's still a true negative.

Messages about user-perceived errors should inform the user as specifically as possible about the system's limitations.





Aim for

Use error states to tell the user what inputs the AI needs or how the AI works. Learn more

Avoid

Don't miss an opportunity to explain your AI system.

Diagnose errors that users don't perceive

Since these errors are invisible to the user, you don't need to worry about how to explain them in the user interface, but being aware of them could help you improve your AI.

Happy accidents. The system might flag something as a poor prediction, limitation, or error, but in rare cases it could be helpful or interesting anyway. For example, a user asks a smart speaker to take out the trash as a joke, knowing it isn't possible, and she and her family are amused at the speaker's response.

Background errors. Situations in which the system isn't working correctly, but neither the user nor the system register an error. For example, if a search engine returns an incorrect result and the user is unable to identify it as such.

These types of errors may have significant consequences if undetected for long periods of time. They also have important implications for how your systems measure failure and error rates. Because you're unlikely to detect these errors from user feedback, you'll need a dedicated quality assurance process to stress test the system and identify "unknown unknowns".

Account for timing in the user journey

Users may react differently to errors they encounter in the early days of using your product versus later, when they have more ingrained expectations of what it can and can't do. How long the user has been using the product should impact how your AI communicates errors and helps users get back on track.

For example, when a user interacts with a music recommendation system for the first time, they might not consider it an error when the initial recommendations aren't relevant to them. However, after a year of listening, liking, and adding favorites to playlists, the user might have higher expectations for the relevance of the system's recommendations and therefore consider irrelevant recommendations to be errors.

Weigh situational stakes & error risk

In some situations, AI errors and failure are inconvenient, but in others, they could have severe consequences. For example, errors in AI-suggested email responses have very different stakes than errors related to autonomous vehicle handling. This is true for non-AI systems as well, but the complexity of AI systems and the possibility of context errors requires that you give extra thought to the stakes of each situation where your AI is making decisions.

Any given scenario has inherent stakes, but the risk of errors can change depending on other contextual factors. For example, if a user is multitasking or is under time pressure while using your AI-driven product, then they have fewer mental resources available to double-check the system output.

Gauge the risk for potential errors

Lower

- User has expertise in the task
- Extremely high system confidence
- Many possible successful outcomes

Higher

- User is a novice at the task
- Reduced attention or response time (multi-tasking)
- Low system confidence
- Narrow definition of successful outcomes

Assess the stakes of the situation

Lower

- Experimentation
- Play or creativity
- Lightweight entertainment
- Non-essential recommendations

Higher

- Health, safety, or financial decisions
- Sensitive social contexts

When considering what system responses, user responses, and messaging are required in different situations, refer to the section on designing your reward function in the User Needs + Defining Success chapter. Refer also to the high-stakes scenarios and corresponding required explanations identified in the Explainability + Trust chapter.

Key concepts

Before you start designing how your system will respond to errors, try to identify errors that your users can perceive already, or that you can predict will occur. Then, sort them into the following categories.

1. **System limitation**. Your system can't provide the right answer, or any answer at all, due to inherent limitations to the system.
2. **Context**. The system is "working as intended," but the user perceives an error because the actions of the system aren't well-explained, break the user's mental model, or were based on poor assumptions.

Also be on the lookout for **background errors**, situations in which the system isn't working correctly, but neither the user nor the system register an error.
For each situation above, ask yourself:

- How does this error impact the user?
- Are the stakes high or low for the user in this situation?

Apply the concepts from this section in Exercise 1 in the worksheet

② Identify error sources

The User Needs + Defining Success chapter illustrates how to design and evaluate the reward function , which the AI uses to optimize its output. The range and type of possible errors in your system will depend on this reward function, but generally speaking there are several sources of errors unique to AI.

Discover prediction & training data errors

These errors come from issues with your training data and the way you've tuned your machine learning model. For example, a navigation app might not have had training data for roads in a certain geographic region, limiting its capabilities.

Mislabeled or misclassified results

When the output from the system is incorrectly labeled or classified due to poor training data. For example, inconsistently-labeled berries in the training data for a plant classification app leads to misclassification of strawberries as raspberries.

Response: Allow users to give guidance or correct the data or label, which feeds back into the model to improve the dataset or alert the team to the need for additional training data.

Poor inference or incorrect model

When the machine learning model isn't sufficiently precise , despite having adequate training data. For example, a plant classification app has a well-labeled dataset that includes berries, but poor model tuning returns a large number of false positives when identifying strawberries.

Response: Allow users to give guidance, or correct the data or label, which feeds back into the AI model and helps you tune it.

Missing or incomplete data

When the user reaches the edges of what the model can do based on the data it was trained on. For example, if a user tries to use a plant classification app on a dog.

Response: Communicate what the system is supposed to do and how it works. Then, explain what it's missing or its limitations. Allow users to give feedback about the needs that the system isn't meeting.

Review the chapters on Data Collection + Evaluation and Explainability + Trust to see more information on how to identify the right data sources and how to explain them.

Predict or plan for input errors

These errors come from the user's expectation that the system will "understand" what they really mean, whether or not the AI is actually capable of doing so.

Unexpected input

When a user anticipates the auto-correction of their input into an AI system. For example, the user makes a typo and expects the system to recognize their intended spelling.

Response: Check the user's input versus a range of "expected" answers to see if they intended one of those inputs. For example, "Did you mean to search for XYZ?"

Breaking habituation

When a user has built up habitual interactions with the system's UI, but a change causes their actions to lead to a different, undesired, result.

For example, in a file storage system, the user has always accessed a folder by clicking in the top right hand region of the interface, but due to newly-implemented AI-driven dynamic design, folder locations change frequently. Clicking in that area due to muscle memory now opens the wrong folder.

Response: Implement AI in ways that don't break habituation, such as by designating a specific area of the interface for less-predictable AI output. Or, allow users to revert to, choose, or retrain a specific interaction pattern. See more suggestions in this article on habituation from the Google Design blog.

Miscalibrated input

When a system improperly weights an action or choice. For example, if one search for an artist in a music app leads to endless recommendations for that artist's music.

Response: Explain how the system matches inputs to outputs and allow the user to correct the system through feedback.

Check output quality for relevance errors

Your AI-driven system won't always be able to provide the appropriate information at the right time. Irrelevance is often the source of context errors — conflicts between the system working as intended and the user's real-life needs. Unlike user input errors, or data errors, these aren't issues with the veracity of the information or how the system makes decisions. These errors are issues in how and when those results are delivered — or not.

Low confidence

When the model can't fulfill a given task due to uncertainty restraints: lack of available data, requirements for prediction accuracy, or unstable information. For example, if a flight price prediction algorithm can't accurately predict next year's prices because of changing conditions.

Response: Explain why a certain result couldn't be given and provide alternative paths forward. For example, "There's not enough data to predict prices for flights to Paris next year. Try checking again in a month".

Irrelevance

When the system output is high confidence but presented to users in a way that isn't relevant to the user's needs. For example, a user books a trip to Houston for a family funeral and their travel app recommends some "fun vacation activities".

Response: Allow the user to provide feedback to improve the system's function.

Disambiguate systems hierarchy errors

These errors arise from using multiple AI systems that may not be communicating with one another. These overlaps can result in conflicts over situational control or a user's attention. Plan ahead for how your AI system might interact with other systems, and provide users with multiple levels of control for managing output and avoiding conflicts.

Multiple systems

When a user connects your product to another system, and it isn't clear which system is in charge at a given time.

For example, a user connects a "smart thermostat" to a "smart energy meter," but the two systems have different methods of optimizing for energy efficiency. The systems send conflicting signals to one another and stop working properly.

Response: Explain the multiple systems that are connected, and allow the user to determine priority. Consider visual ways to represent the relationship between multiple AI systems in the product interface, perhaps by mapping them onto different locations.

Signal crashes

When multiple systems are monitoring a single (or similar) outputs and an event causes simultaneous alerts.

Signal crashes increase the user's mental load because the user has to parse multiple information sources to figure out what happened, and what action they need to take.

For example, imagine if a user's voice input was recognized by multiple voice-activated systems, which simultaneously respond in different ways. That would be overwhelming, and the user likely wouldn't know what to do next.

Response: Allow the user to set independent controls for your AI that don't overlap with other signals. For example, the watch word for a smart speaker to start listening could be unique. If your team can avoid creating new context errors, the system could attempt to infer which system is the one the user intended to have primacy.

Key concepts

For each error, try to determine the cause based on the resulting user experience. Ask yourself what kind of error it could be based on the examples below.

Prediction & training data errors occur when… the system's capabilities are limited by its training data.
Input errors occur when… the user anticipates the auto-correction of their input into an AI system, or their habituation is interrupted.
Relevance errors occur when… the system output appears in a time, place, or format that isn't relevant to the user's needs.
System hierarchy errors occur when… your user connects your product to another system, and it isn't clear which system is in charge.

Apply the concepts from this section in Exercise 2 in the worksheet

③ Provide paths forward from failure

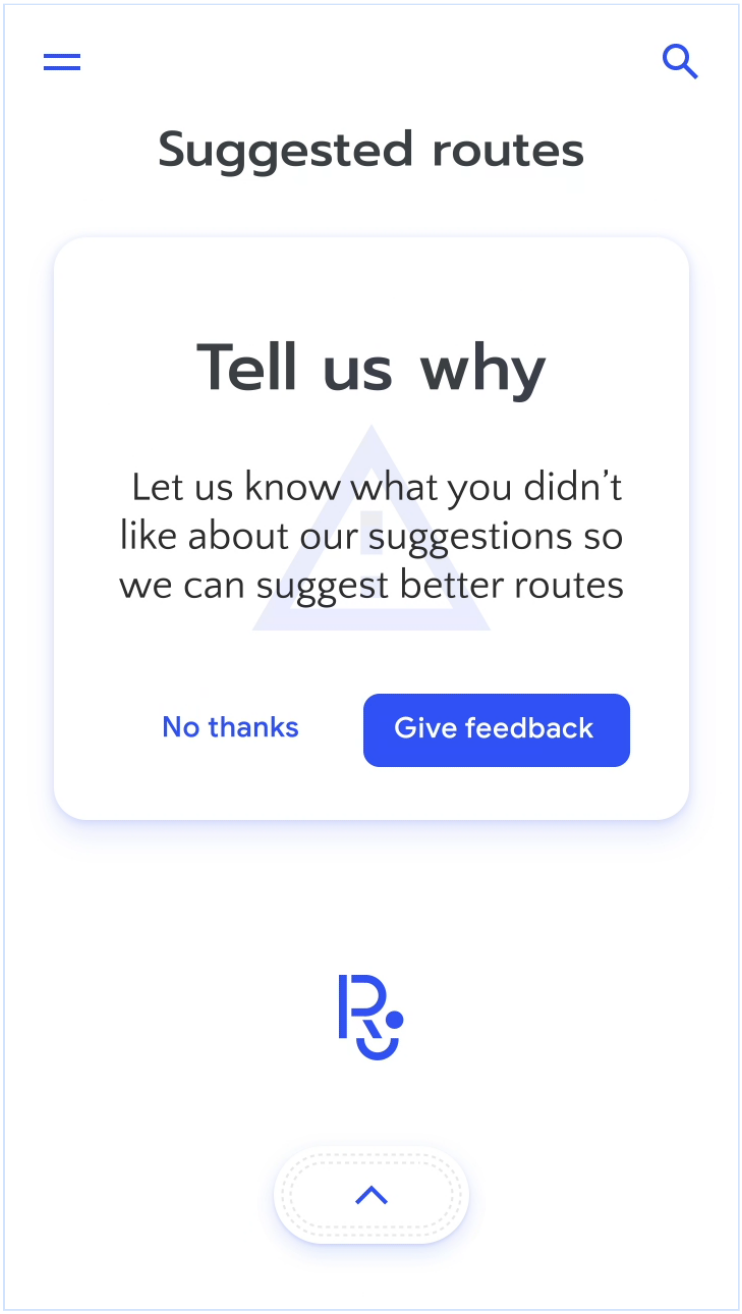
Setting reasonable expectations about the limitations of your technology helps set up good experiences with errors, but how users move forward is equally important. Focusing on what your users can do after the system fails empowers them while maintaining the usefulness of your product.

If you were a waiter in a restaurant, and a customer wanted something on the menu but the kitchen ran out, you'd have come up with an alternative option, taking into account their implied preferences. For example, "We don't have Coke, but is Pepsi OK?" In the case of higher situational stakes, there should be an additional check to make sure the alternative is indeed a safe option. For example, if the user wants a dish the restaurant doesn't have, but the closest alternative contains a common allergen, the waiter would want to think twice before making the recommendation. Your primary goal in these failstates should be to prevent undue harm to the user and help them move forward with their task.

Create opportunities for feedback

As highlighted in the above sections, many of the errors your users experience require their feedback in order to improve the system. In particular, error types that aren't easily recognized by the system itself, such as mislabeled data, rely on outside feedback to fix.

Include opportunities for users to provide feedback both when presented with an error message as well as alongside "correct" system output. For example, ask a user what they expected to happen after a system failure, and provide the option to report inappropriate suggestions.



Aim for

Ask the user for feedback if they repeatedly reject AI outputs. Learn more

Return control to the user

When an AI system fails, often the easiest path forward is to let the user take over. Whether or not giving users full “override” powers is possible and what it looks like will differ from system to system. In some cases, reverting to manual control could be risky or dangerous. For example, if a navigation system stops giving directions altogether to a user who is navigating through an unfamiliar environment during rush hour traffic.

When this AI-to-manual control transition happens, it’s your responsibility to make it easy and intuitive for users to quickly pick up where the system leaves off. That means the user must have all the information they need in order to take the reins: awareness of the situation, what they need to do next, and how to do it.

Assume subversive use

Though it’s important to write informative and actionable error messages, your team should design your product knowing that some people will intentionally abuse it. That means that in addition to mapping out potential negative impacts of your product, like in the User Needs + Defining Success chapter, you should try to make failure safe, boring, and a natural part of the product. Avoid making dangerous failures interesting, or over-explaining system vulnerabilities — that can incentivize the users to reproduce them.

For example, if every time a user marked an email in their inbox as spam, the email app explained why the system did not classify that message as spam, that could give spammers useful tips on how to avoid getting caught by the filter.

To get a really clear idea of all the potential dead-ends in your product, invest in beta-testing and pilot programs. Many products are like a maze of options and possibilities; while the builders focus on the happy path, users can get caught in the labyrinth.

Key concepts

You can use quality assurance exercises and pilots initially to find errors, but you should continue monitoring to cover any new features you launch. As a team, decide on some channels you'll monitor for new error discoveries from your users. Which of the following error report sources could work for your team?

- Reports sent to customer service
- Comments and reports sent through social media channels
- In-product metrics
- In-product surveys
- User research
    - Out-of-product surveys
    - Deep-dive interviews
    - Diary studies

As you discover each error, you'll then need to work as a team to design the correct path forward from that error.

Apply the concepts from this section in Exercise 2 in the worksheet

Summary

Learning, machine or otherwise, can't happen without making mistakes. Designing and building your system with the knowledge that errors are integral will help you create opportunities for dialogue with your users. This in turn creates more efficient pathways for error resolution, and for the users to complete their goals.

When designing your error experience, be human, not machine. Error messages may need to disclose that the system made a mistake. Address mistakes with humanity and humility, and explain the system's limits while inviting people to continue forward.

① **Define "errors" & "failure"**. When dealing with a probabilistic , dynamic system, a user could perceive a failure in situations where the system is working as intended. Acknowledging that a product is a work-in-progress can help encourage the adoption and feedback that designers and engineers need to continue improving the AI.

② **Identify error sources**. The inherent complexity of AI-powered systems can make identifying the source of an error challenging. It's important to discuss as a team how you'll discover errors and discern their sources.

③ **Provide paths forward from failure**. The trick isn't to avoid failure, but to find it and make it just as user-centered as the rest of your product. No matter how hard you work to ensure a well-functioning system, AI is probabilistic by nature, and like all systems, will fail at some point. When this happens, the product needs to provide ways for the user to continue their task and to help the AI improve.

Want to drive discussions, speed iteration, and avoid pitfalls? Use the worksheet